Optimization Problem Types - Mixed-Integer and Constraint Programming

Optimization Problem Types

Mixed-Integer Programming (MIP) Problems

A mixed-integer programming (MIP) problem is one where some of the **decision variables** are constrained to have only **integer values** (i.e. whole numbers such as -1, 0, 1, 2, etc.) at the optimal solution. The use of integer variables greatly expands the scope of useful optimization problems that you can be define and solve.

An important special case is a decision variable x1 that is integer with 0 <= x1 <= 1. This forces x1 to be either 0 or 1 at the solution. Variables like x1, called **0-1 or binary integer variables**, can be used to model yes/no decisions, such as whether to build a plant or buy a piece of equipment.

However, integer variables make an optimization problem **far more difficult to solve**. Memory and solution time may rise **exponentially** as you add more integer variables. Even with highly sophisticated algorithms and modern supercomputers, there are models of just a few hundred integer variables that have never been solved to optimality.

This is because many combinations of specific integer values for the variables must be tested, and **each combination requires the solution of a "normal" linear or nonlinear optimization problem**. The number of combinations can rise exponentially with the size of the problem.

Constraint Programming (CP) Problems

The term constraint programming comes from artificial intelligence research, where there are many problems that require assignment of symbolic values (such as positions on a chessboard) to variables that satisfy certain constraints. The symbolic values come from a finite set of possibilities, and these possibilities can be numbered with integers.

Constraint programming defines "higher-level" constraints that apply to integer variables. The most common and useful higher-level constraint is the **alldifferent constraint**, which applies to a set of variables, say x1, x2, x3, x4 and x5. This constraint assumes that the variables can have only a finite number of possible values (say 1 through 5), and specifies that the variables must be all different at the optimal solution.

Values such as 1, 2, 3, 4, 5 or 5, 4, 3, 2, 1 for the variables would satisfy this constraint, but any assignment of the same value to two or more different variables (e.g. 1, 2, 3, 1, 4) would violate the alldifferent constraint. Thus, the assignment must be an **ordering or permutation** of the integers 1 through 5.

A classic example of a constraint programming problem is the **traveling salesman problem**: A salesman plans to visit N cities and must drive varying distances between them. In what order should he/she visit the cities to minimize the total distance traveled, while visiting each city exactly once?

Constraint programming problems have all the advantages and disadvantages of mixed-integer programming problems, and the extra requirements such as "alldifferent" generally make such problems even harder to solve. All of Frontline's solvers support the alldifferent constraint, but you must bear in mind the implications for solution time if you use such constraints.

Solving MIP and CP Problems

The "classic" method for solving MIP and CP problems is called **Branch and Bound**. This method begins by finding the optimal solution to the "relaxation" of the problem without the integer constraints (via standard linear or nonlinear optimization methods). If in this solution, the decision variables with integer constraints have integer values, then no further work is required. If one or more integer variables have non-integral solutions, the Branch and Bound method chooses one such variable and "branches," creating two new subproblems where the value of that variable is more tightly constrained. These subproblems are solved and the process is repeated, until a solution that satisfies all of the integer constraints is found.

Alternative methods, such as genetic and evolutionary algorithms, randomly generate candidate solutions that satisfy the integer constraints. Such initial solutions are usually far from optimal, but these methods then transform existing solutions into new candidate solutions, through methods such as **integer- or permutation-preserving mutation and crossover**, that continue to satisfy the integer constraints, but may have better objective values. This process is repeated until a sufficiently "good solution" is found. Generally, these methods are not able to "prove optimality" of the solution.

Frontline Systems Solver Technology and Products for MIP and CP Problems

Other Problem Types

About Us | Contact Us | Search